WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Friday, August 03, 2007

Hide?	Set Name	Query	Hit Count
	DB=PGPB,USPT,USOC; PLUR=YES; OP=ADJ		
☐	L23	L22 and trap.clm.	0
☐	L22	L17 and (kernel).clm.	6
☐	L21	L17 and (look up table).clm.	1
☐	L20	L17 and (lookup table).clm.	1
☐	L19	L17 and (look-up table).clm.	1
☐	L18	L17 and (program counter).clm.	0
☐	L17	L16 and (trace.clm. or tracing.clm.)	942
☐	L16	probe.clm.	61940
☐	L15	L14 and (program counter).clm.	1
☐	L14	(scratch space).clm.	11
	DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ		
☐	L13	L6 and (anonymous near(trace or tracing))	4
☐	L12	L11 and kernel	20
☐	L11	L7 and probe	50
☐	L10	L7 and (trap near (handler or handling))	28
☐	L9	L8 and (trap near (handler or handling))	0
☐	L8	L6 and placeholder	246
☐	L7	((trace or tracing) and instrument$) near2 (program or application)	302
☐	L6	(trace or tracing) and instrument$	69045
☐	L5	L3 and anonymous	2
☐	L4	L3 and kernel	21
☐	L3	L2 and probe	83
☐	L2	L1 and (trace or tracing) and instrument$	441
☐	L1	(717/124 |717/125 |717/126 |717/127 |717/128 |717/129 |717/130 |717/131 |717/132 |717/133 |717/134 |717/135).ccls.	3278

Interference search

END OF SEARCH HISTORY

WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Friday, August 03, 2007

Hide?	Set Name	Query	Hit Count
	DB=PGPB,USPT,USOC; PLUR=YES; OP=ADJ		
☐	L24	L21 and (scratch).clm.	0
☐	L23	L21 and (trap handler).clm.	0
☐	L22	L21 and (trap instruction).clm.	0
☐	L21	L20 and thread.clm.	313
☐	L20	trace.clm. or tracing.clm.	30655
☐	L19	(trace).clm. or tracing.clm.	30655
☐	L18	L14 and (trace).clm.	0
☐	L17	L14 and (tracing).clm.	0
☐	L16	L14 and (trap).clm.	0
☐	L15	L14 and (program counter).clm.	1
☐	L14	(scratch space).clm.	11
	DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=ADJ		
☐	L13	L6 and (anonymous near(trace or tracing))	4
☐	L12	L11 and kernel	20
☐	L11	L7 and probe	50
☐	L10	L7 and (trap near (handler or handling))	28
☐	L9	L8 and (trap near (handler or handling))	0
☐	L8	L6 and placeholder	246
☐	L7	((trace or tracing) and instrument$) near2 (program or application)	302
☐	L6	(trace or tracing) and instrument$	69045
☐	L5	L3 and anonymous	2
☐	L4	L3 and kernel	21
☐	L3	L2 and probe	83
☐	L2	L1 and (trace or tracing) and instrument$	441
☐	L1	(717/124 |717/125 |717/126 |717/127 |717/128 |717/129 |717/130 |717/131 |717/132 |717/133 |717/134 |717/135).ccls.	3278

Interference search

END OF SEARCH HISTORY

Terms used: **scratch space tracing instrument** Found **8** of **207,474**

Sort results by: relevance
Display results: expanded form
Save results to a Binder
Search Tips
Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 8 of 8

Relevance scale

1 Maintaining Consistency and Bounding Capacity of Software Code Caches
Derek Bruening, Saman Amarasinghe
March 2005 **Proceedings of the international symposium on Code generation and optimization CGO '05**
Publisher: IEEE Computer Society
Full text available: pdf(253.56 KB) Additional Information: full citation, abstract, citings, index terms

Software code caches are becoming ubiquitous, in dynamic optimizers, runtime tool platforms, dynamic translators, fast simulators and emulators, and dynamic compilers. Caching frequently executed fragments of code provides significant performance boosts, reducing the overhead of translation and emulation and meeting or exceeding native performance in dynamic optimizers. One disadvantage of caching, memory expansion, can sometimes be ignoredwhen executing a single application. However, as optimiz ...

2 On the usefulness of type and liveness accuracy for garbage collection and leak detection
Martin Hirzel, Amer Diwan, Johannes Henkel
November 2002 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 24 Issue 6
Publisher: ACM Press
Full text available: pdf(684.85 KB) Additional Information: full citation, abstract, references, citings, index terms

The effectiveness of garbage collectors and leak detectors in identifying dead objects depends on the *accuracy* of their reachability traversal. Accuracy has two orthogonal dimensions: (i) whether the reachability traversal can distinguish between pointers and nonpointers (*type accuracy*), and (ii) whether the reachability traversal can identify memory locations that will be dereferenced in the future (*liveness accuracy*). This article presents an experimental study of the impo ...

Keywords: Conservative garbage collection, leak detection, liveness accuracy, program analysis, type accuracy

3 RaceTrack: efficient detection of data race conditions via adaptive tracking
Yuan Yu, Tom Rodeheffer, Wei Chen
October 2005 **ACM SIGOPS Operating Systems Review , Proceedings of the twentieth ACM symposium on Operating systems principles SOSP '05**, Volume 39 Issue 5
Publisher: ACM Press
Full text available: pdf(321.34 KB) Additional Information: full citation, abstract, references, citings, index terms

Bugs due to data races in multithreaded programs often exhibit non-deterministic symptoms and are notoriously difficult to find. This paper describes RaceTrack, a dynamic race detection tool that tracks the actions of a program and reports a warning whenever a suspicious pattern of activity has been observed. RaceTrack uses a novel hybrid detection algorithm and employs an adaptive approach that automatically directs more effort to areas that are more suspicious, thus providing more accurate war ...

Keywords: race detection, virtual machine instrumentation

4 Memory: Coupling prefix caching and collective downloads for remote dataset access

Xiaosong Ma, Vincent W. Freeh, Tao Yang, Sudharshan S. Vazhkudai, Tyler A. Simon, Stephen L. Scott

June 2006 **Proceedings of the 20th annual international conference on Supercomputing ICS '06**

Publisher: ACM Press

Full text available: pdf(490.30 KB) Additional Information: full citation, abstract, references, cited by, index terms

Scientific datasets are typically archived at mass storage systems or data centers close to supercomputers/instruments. End-users of these datasets, however, usually perform parts of their workflows at their local computers. In such cases, client-side caching can offer significant gains by reducing the cost of wide-area data movement.Scientific data caches, however, traditionally cache entire data-sets, which may not be necessary. In this paper, we propose a novel combination of prefix caching < ...

5 Constructing collaborative desktop storage caches for large scientific datasets

Sudharshan S. Vazhkudai, Xiaosong Ma, Vincent W. Freeh, Jonathan W. Strickland, Nandan Tammineedi, Tyler Simon, Stephen L. Scott

August 2006 **ACM Transactions on Storage (TOS)**, Volume 2 Issue 3

Publisher: ACM Press

Full text available: pdf(833.76 KB) Additional Information: full citation, abstract, references, index terms

High-end computing is suffering a *data deluge* from experiments, simulations, and apparatus that creates overwhelming application dataset sizes. This has led to the proliferation of high-end mass storage systems, storage area clusters, and data centers. These storage facilities offer a large range of choices in terms of capacity and access rate, as well as strong data availability and consistency support. However, for most end-users, the "last mile" in their analysis pipeline o ...

Keywords: Distributed storage, parallel I/O, scientific data management, serverless storage system, storage cache, storage networking, storage resoucce management, storage scavenging, striped storage

6 Proceedings of the SIGNUM conference on the programming environment for development of numerical software

March 1979 **ACM SIGNUM Newsletter**, Volume 14 Issue 1

Publisher: ACM Press

Full text available: pdf(5.02 MB) Additional Information: full citation

7 Performance characterization of a Quad Pentium Pro SMP using OLTP workloads

Kimberly Keeton, David A. Patterson, Yong Qiang He, Roger C. Raphael, Walter E. Baker

April 1998 **ACM SIGARCH Computer Architecture News , Proceedings of the 25th annual international symposium on Computer architecture ISCA '98**, Volume 26 Issue 3

Publisher: IEEE Computer Society, ACM Press

Full text available: pdf(1.58 MB) Additional Information: full citation, abstract, references, citings, index

Publisher Site terms

Commercial applications are an important, yet often overlooked, workload with significantly different characteristics from technical workloads. The potential impact of these differences is that computers optimized for technical workloads may not provide good performance for commercial applications, and these applications may not fully exploit advances in processor design. To evaluate these issues, we use hardware counters to measure architectural features of a four-processor Pentium Pro-based se ...

8 The Vesta parallel file system

Peter F. Corbett, Dror G. Feitelson

August 1996 **ACM Transactions on Computer Systems (TOCS)**, Volume 14 Issue 3

Publisher: ACM Press

Full text available: pdf(649.08 KB) Additional Information: full citation, abstract, references, citings, index terms, review

The Vesta parallel file system is designed to provide parallel file access to application programs running on multicomputers with parallel I/O subsystems. Vesta uses a new abstraction of files: a file is not a sequence of bytes, but rather it can be partitioned into multiple disjoint sequences that are accessed in parallel. The partitioning—which can also be changed dynamically—reduces the need for synchronization and coordination during the access. Some control over the layout ...

Keywords: data partitioning, parallel computing, parallel file system

Results 1 - 8 of 8

Terms used: **scratch space tracing trap** Found **4** of **207,474**

Sort results by: relevance | Display results: expanded form | Save results to a Binder | Search Tips | Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 4 of 4

Relevance scale

1 Maintaining Consistency and Bounding Capacity of Software Code Caches
Derek Bruening, Saman Amarasinghe
March 2005 **Proceedings of the international symposium on Code generation and optimization CGO '05**
Publisher: IEEE Computer Society
Full text available: pdf(253.56 KB) Additional Information: full citation, abstract, citings, index terms

Software code caches are becoming ubiquitous, in dynamic optimizers, runtime tool platforms, dynamic translators, fast simulators and emulators, and dynamic compilers. Caching frequently executed fragments of code provides significant performance boosts, reducing the overhead of translation and emulation and meeting or exceeding native performance in dynamic optimizers. One disadvantage of caching, memory expansion, can sometimes be ignoredwhen executing a single application. However, as optimiz ...

2 Proceedings of the SIGNUM conference on the programming environment for development of numerical software
March 1979 **ACM SIGNUM Newsletter**, Volume 14 Issue 1
Publisher: ACM Press
Full text available: pdf(5.02 MB) Additional Information: full citation

3 Binary translation
Richard L. Sites, Anton Chernoff, Matthew B. Kirk, Maurice P. Marks, Scott G. Robinson
February 1993 **Communications of the ACM**, Volume 36 Issue 2
Publisher: ACM Press
Full text available: pdf(4.84 MB) Additional Information: full citation, references, citings, index terms

Keywords: CISC computers, RISC computers, binary translation, computer architecture, processor architecture translation

4 DFTI---a new interface for Fast Fourier Transform libraries
Ping Tak Peter Tang
December 2005 **ACM Transactions on Mathematical Software (TOMS)**, Volume 31 Issue 4
Publisher: ACM Press
Full text available: pdf(211.61 KB) Additional Information: full citation, abstract, references, index terms

The Fast Fourier Transform (FFT) algorithm that calculates the Discrete Fourier Transform (DFT) is one of the major breakthroughs in scientific computing and is now an

indispensable tool in a vast number of fields. Unfortunately, software applications that provide fast computation of DFT via FFT differ vastly in functionality and lack uniformity. A widely accepted Applications Programmer Interface (API) for DFT would advance the field of scientific computing significantly. In this article, we pr ...

Keywords: API, FFT, interface, software

Results 1 - 4 of 4

Terms used: **kernel tracing trap instrument program counter** Found **32** of **207,474**

Sort results by: relevance
Display results: expanded form
Save results to a Binder
Search Tips
Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 32 Result page: **1** 2 next

Relevance scale

1 Advanced aspects: A dynamic aspect-oriented system for OS kernels

Yoshisato Yanagisawa, Kenichi Kourai, Shigeru Chiba
October 2006 **Proceedings of the 5th international conference on Generative programming and component engineering GPCE '06**
Publisher: ACM Press
Full text available: pdf(248.81 KB) Additional Information: full citation, abstract, references, index terms

We propose a dynamic aspect-oriented system for operating system (OS) kernels written in the C language. Unlike other similar systems, our system named *KLASY* allows the users to pointcut not only function calls but also member accesses to structures. This feature helps the developers who want to use aspects for profiling or debugging an OS kernel. To enable this, KLASY uses a modified C compiler for compiling an OS kernel. The modified compiler produces extended symbol information, which ...

Keywords: Linux, aspect-oriented programming, dynamic AOP, operating system, profiling and debugging

2 Trace-driven memory simulation: a survey

Richard A. Uhlig, Trevor N. Mudge
June 1997 **ACM Computing Surveys (CSUR)**, Volume 29 Issue 2
Publisher: ACM Press
Full text available: pdf(636.11 KB) Additional Information: full citation, abstract, references, citings, index terms, review

As the gap between processor and memory speeds continues to widen, methods for evaluating memory system designs before they are implemented in hardware are becoming increasingly important. One such method, trace-driven memory simulation, has been the subject of intense interest among researchers and has, as a result, enjoyed rapid development and substantial improvements during the past decade. This article surveys and analyzes these developments by establishing criteria for evaluating trac ...

Keywords: TLBs, caches, memory management, memory simulation, trace-driven simulation

3 Active memory: a new abstraction for memory system simulation

Alvin R. Lebeck, David A. Wood
January 1997 **ACM Transactions on Modeling and Computer Simulation (TOMACS)**, Volume 7 Issue 1
Publisher: ACM Press
Full text available: pdf(690.38 KB) Additional Information: full citation, references, citings, index terms

Keywords: Cache memory, direct-execution simulation, memory hierarchy, on-the-fly simulation, trace-driven simulation

4 Valgrind: a framework for heavyweight dynamic binary instrumentation

Nicholas Nethercote, Julian Seward

June 2007 **ACM SIGPLAN Notices , Proceedings of the 2007 ACM SIGPLAN conference on Programming language design and implementation PLDI '07**, Volume 42 Issue 6

Publisher: ACM Press

Full text available: pdf(236.38 KB) Additional Information: full citation, abstract, references, index terms

Dynamic binary instrumentation (DBI) frameworks make it easy to build dynamic binary analysis (DBA) tools such as checkers and profilers. Much of the focus on DBI frameworks has been on performance; little attention has been paid to their capabilities. As a result, we believe the potential of DBI has not been fully exploited.

In this paper we describe Valgrind, a DBI framework designed for building heavyweight DBA tools. We focus on its unique support for *shadow values*-a powerfu ...

Keywords: Memcheck, Valgrind, dynamic binary analysis, dynamic binary instrumentation, shadow values

5 Using the SimOS machine simulator to study complex computer systems

Mendel Rosenblum, Edouard Bugnion, Scott Devine, Stephen A. Herrod

January 1997 **ACM Transactions on Modeling and Computer Simulation (TOMACS)**, Volume 7 Issue 1

Publisher: ACM Press

Full text available: pdf(731.76 KB) Additional Information: full citation, references, citings, index terms, review

Keywords: computer architecture, computer simulation, computer system performance analysis, operating systems

6 A comparison of software and hardware techniques for x86 virtualization

Keith Adams, Ole Agesen

October 2006 **ACM SIGOPS Operating Systems Review , ACM SIGARCH Computer Architecture News , ACM SIGPLAN Notices , Proceedings of the 12th international conference on Architectural support for programming languages and operating systems ASPLOS-XII**, Volume 40 , 34 , 41 Issue 5 , 5 , 11

Publisher: ACM Press

Full text available: pdf(236.66 KB) Additional Information: full citation, abstract, references, index terms

Until recently, the x86 architecture has not permitted classical trap-and-emulate virtualization. Virtual Machine Monitors for x86, such as VMware ® Workstation and Virtual PC, have instead used binary translation of the guest kernel code. However, both Intel and AMD have now introduced architectural extensions to support classical virtualization.We compare an existing software VMM with a new VMM designed for the emerging hardware support. Surprisingly, the hardware VMM often suffers lower ...

Keywords: MMU, SVM, TLB, VT, dynamic binary translation, nested paging, virtual machine monitor, virtualization, x86

7

Security and eliability: Using VMM-based sensors to monitor honeypots
Kurniadi Asrigo, Lionel Litty, David Lie
June 2006 **Proceedings of the second international conference on Virtual execution environments VEE '06**
Publisher: ACM Press
Full text available: pdf(232.05 KB) Additional Information: full citation, abstract, references, index terms

Virtual Machine Monitors (VMMs) are a common tool for implementing honeypots. In this paper we examine the implementation of a VMM-based intrusion detection and monitoring system for collecting information about attacks on honeypots. We document and evaluate three designs we have implemented on two open-source virtualization platforms: User-Mode Linux and Xen. Our results show that our designs give the monitor good visibility into the system and thus, a small number of monitoring sensors can det ...

Keywords: IDS, honeypot monitoring, intrusion detection, virtual machine monitor

8 Workshop on Dynamic Analysis (WODA): Merging traces of hardware-assisted data breakpoints
Mayur Palankar, Jonathan E. Cook
May 2005 **ACM SIGSOFT Software Engineering Notes , Proceedings of the third international workshop on Dynamic analysis WODA '05**, Volume 30 Issue 4
Publisher: ACM Press
Full text available: pdf(158.06 KB) Additional Information: full citation, abstract, references, citings, index terms

Future improvements in hardware and O/S support for monitoring programs will depend on providing feedback for current support (even if it is quite limited). We look at using hardware breakpoint registers and performance counters in order to trace data accesses in a program. We first present a small experiment to understand how these features can be used to monitor a program. and then detail an algorithm for using these limited resources to trace any amount of data accesses in a program and achie ...

9 The impact of architectural trends on operating system performance
M. Rosenblum, E. Bugnion, S. A. Herrod, E. Witchel, A. Gupta
December 1995 **ACM SIGOPS Operating Systems Review , Proceedings of the fifteenth ACM symposium on Operating systems principles SOSP '95**, Volume 29 Issue 5
Publisher: ACM Press
Full text available: pdf(2.03 MB) Additional Information: full citation, references, citings, index terms

10 WBIA'05: Transparent debugging of dynamically instrumented programs
Naveen Kumar, Ramesh Peri
December 2005 **ACM SIGARCH Computer Architecture News**, Volume 33 Issue 5
Publisher: ACM Press
Full text available: pdf(280.45 KB) Additional Information: full citation, abstract, references, index terms

Dynamic instrumentation systems, used for program analysis, bug isolation, software security and simulations, are becoming increasingly popular. There exists a need to debug dynamically instrumented programs while keeping the presence of dynamic instrumentation system hidden from debug users. Existing debuggers use debug information in program binaries that have been generated by a compiler at static compile time, to provide their debug support. Since dynamic instrumentation systems generate pro ...

11 Architectural support for software-based protection
Mihai Budiu, Úlfar Erlingsson, Martín Abadi
October 2006 **Proceedings of the 1st workshop on Architectural and system support for improving software dependability ASID '06**

Publisher: ACM Press

Full text available: pdf(642.62 KB) Additional Information: full citation, abstract, references, index terms

Control-Flow Integrity (CFI) is a property that guarantees program control flow cannot be subverted by a malicious adversary, even if the adversary has complete control of data memory. We have shown in prior work how CFI can be enforced by using inlined software guards that perform safety checks. The first part of this paper shows how modest Instruction Set Architecture (ISA) support can replace such guard code with single instructions.On the foundation of CFI we have implemented XFI: a protecti ...

Keywords: binary rewriting, control-flow graph, control-flow integrity, hardware support, memory protection, security, software fault isolation

12 Language-based security: Dynamic label binding at run-time

Yolanta Beres, Chris I. Dalton

August 2003 **Proceedings of the 2003 workshop on New security paradigms NSPW '03**

Publisher: ACM Press

Full text available: pdf(727.41 KB) Additional Information: full citation, abstract, references

Information flow control allows enforcement of end-to-end confidentiality policies but has been difficult to put in practice. This paper introduces a pragmatic new approach for tracking information flow while the process is running at the same time applying dynamic label binding. The underlying implementation mechanism uses machine code instruction stream modification to track individual data movements and manipulations within the address space of an application. This gives the ability to precis ...

Keywords: data labeling, information flow control, labels

13 Fast data-locality profiling of native execution

Erik Berg, Erik Hagersten

June 2005 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems SIGMETRICS '05**, Volume 33 Issue 1

Publisher: ACM Press

Full text available: pdf(349.73 KB) Additional Information: full citation, abstract, references, citings, index terms

Performance tools based on hardware counters can efficiently profile the cache behavior of an application and help software developers improve its cache utilization. Simulator-based tools can potentially provide more insights and flexibility and model many different cache configurations, but have the drawback of large run-time overhead.We present StatCache, a performance tool based on a statistical cache model. It has a small run-time overhead while providing much of the flexibility of simulator ...

Keywords: cache behavior, profiling tool

14 Virtual machines: ReVirt: enabling intrusion analysis through virtual-machine logging and replay

George W. Dunlap, Samuel T. King, Sukru Cinar, Murtaza A. Basrai, Peter M. Chen

December 2002 **ACM SIGOPS Operating Systems Review**, Volume 36 Issue SI

Publisher: ACM Press

Full text available: pdf(1.56 MB) Additional Information: full citation, abstract, references, cited by, index terms

Current system loggers have two problems: they depend on the integrity of the operating system being logged, and they do not save sufficient information to replay and analyze attacks that include any non-deterministic events. ReVirt removes the dependency on the

target operating system by moving it into a virtual machine and logging below the virtual machine. This allows ReVirt to replay the system's execution before, during, and after an intruder compromises the system, even if the intruder rep ...

15 Threads and input/output in the synthesis kernal

H. Massalin, C. Pu
November 1989 **ACM SIGOPS Operating Systems Review , Proceedings of the twelfth ACM symposium on Operating systems principles SOSP '89**, Volume 23 Issue 5
Publisher: ACM Press

Full text available: pdf(1.34 MB) Additional Information: full citation, abstract, references, citings, index terms

The Synthesis operating system kernel combines several techniques to provide high performance, including kernel code synthesis, fine-grain scheduling, and optimistic synchronization. Kernel code synthesis reduces the execution path for frequently used kernel calls. Optimistic synchronization increases concurrency within the kernel. Their combination results in significant performance improvement over traditional operating system implementations. Using hardware and software emulating a SUN 3 ...

16 Shade: a fast instruction-set simulator for execution profiling

Bob Cmelik, David Keppel
May 1994 **ACM SIGMETRICS Performance Evaluation Review , Proceedings of the 1994 ACM SIGMETRICS conference on Measurement and modeling of computer systems SIGMETRICS '94**, Volume 22 Issue 1
Publisher: ACM Press

Full text available: pdf(1.28 MB) Additional Information: full citation, abstract, references, citings, index terms

Tracing tools are used widely to help analyze, design, and tune both hardware and software systems. This paper describes a tool called Shade which combines efficient instruction-set simulation with a flexible, extensible trace generation capability. Efficiency is achieved by dynamically compiling and caching code to simulate and trace the application program. The user may control the extent of tracing in a variety of ways; arbitrarily detailed application state information may be collected ...

17 An analysis of operating system behavior on a simultaneous multithreaded architecture

Joshua A. Redstone, Susan J. Eggers, Henry M. Levy
November 2000 **ACM SIGARCH Computer Architecture News , ACM SIGOPS Operating Systems Review , Proceedings of the ninth international conference on Architectural support for programming languages and operating systems ASPLOS-IX**, Volume 28 , 34 Issue 5 , 5
Publisher: ACM Press

Full text available: pdf(227.80 KB) Additional Information: full citation, abstract, references, citings, index terms, review

This paper presents the first analysis of operating system execution on a simultaneous multithreaded (SMT) processor. While SMT has been studied extensively over the past 6 years, previous research has focused entirely on user-mode execution. However, many of the applications most amenable to multithreading technologies spend a significant fraction of their time in kernel code. A full understanding of the behavior of such workloads therefore requires execution and measurement of the operating sy ...

18 An analysis of operating system behavior on a simultaneous multithreaded architecture

Joshua A. Redstone, Susan J. Eggers, Henry M. Levy
November 2000 **ACM SIGPLAN Notices**, Volume 35 Issue 11
Publisher: ACM Press

Full text available: pdf(1.56 MB) Additional Information: full citation, abstract, references, index terms

This paper presents the first analysis of operating system execution on a simultaneous multithreaded (SMT) processor. While SMT has been studied extensively over the past 6 years, previous research has focused entirely on user-mode execution. However, many of the applications most amenable to multithreading technologies spend a significant fraction of their time in kernel code. A full understanding of the behavior of such workloads therefore requires execution and measurement of the operating sy ...

19 Non-intrusive and interactive profiling in parasight

Ziya Aral, Ilya Gertner
January 1988 **ACM SIGPLAN Notices , Proceedings of the ACM/SIGPLAN conference on Parallel programming: experience with applications, languages and systems PPEALS '88**, Volume 23 Issue 9
Publisher: ACM Press

Full text available: pdf(1.05 MB) Additional Information: full citation, abstract, references, citings, index terms

Debugging the performance of parallel applications is crucial for fully utilizing the potential of multiprocessor hardware. This paper describes profiling tools in Parasight, a programming environment that is geared towards non-intrusive performance analysis and high-level debugging. In Parasight, profilers execute as observer programs which run concurrently with the target program and monitor its behavior. This design grew out of our experience in debugging and monitoring the performance o ...

20 DISE: a programmable macro engine for customizing applications

Marc L. Corliss, E. Christopher Lewis, Amir Roth
May 2003 **ACM SIGARCH Computer Architecture News , Proceedings of the 30th annual international symposium on Computer architecture ISCA '03**, Volume 31 Issue 2
Publisher: ACM Press

Full text available: pdf(335.30 KB) Additional Information: full citation, abstract, references, citings

Dynamic Instruction Stream Editing (DISE) is a cooperative software-hardware scheme for efficiently adding customization functionality---e.g, safety/security checking, profiling, dynamic code decompression, and dynamic optimization---to an application. In DISE, application customization functions (ACFs) are formulated as rules for macro-expanding certain instructions into parameterized instruction sequences. The processor executes the rules on the fetched instructions, feeding the executi ...

Results 1 - 20 of 32 Result page: **1** 2 next

Google "scratch space" thread tracing instrument "trap Search Advanced Search Preferences

New! View and manage your web history

Web Results **1** - **10** of about **86** for **"scratch space" thread tracing instrument "trap handler"** . (**0.27** seconds)

Proceedings of the Third Symposium on Operating Systems Design and ...
This will immediately jump to a **trap handler**, which, if it can be safely .... The springboard approach requires chunks of **scratch space** (collectively, ...
www.usenix.org/publications/library/proceedings/osdi99/full_papers/tamches/tamches_html/tamches.html - Similar pages

[PS] Fine-Grained Dynamic Instrumentation of Commodity Operating System ...
File Format: Adobe PostScript - View as Text
This will immediately jump to a **trap handler**, which, if it. can be safely instrumented, ... able **scratch space**: the initialization and termination ...
ftp://ftp.cs.wisc.edu/paradyn/papers/Tamches99FineGrained.ps - Similar pages

[PS] F -G D I C O S K A T A dissertation submitted in partial ...
File Format: Adobe PostScript
if the expense of a stack back-**trace** per blocked **thread** on each sample were not prohibitive, ... tion, which will transfer control to a **trap handler**. ...
ftp://ftp.cs.wisc.edu/paradyn/papers/Tamches01Dissertation.ps - Similar pages
[More results from ftp://ftp.cs.wisc.edu]

Design Doc at OpenSolaris.org
To do this, we introduce a per-brand **scratch space** in the ulwp_t structure, similar to that used by the DTrace PID provider. When we want to **trace** a system ...
www.opensolaris.org/os/community/brandz/design/;jsessionid=08080D0C7080A71D5B0AA3952499A170 - 126k - Cached - Similar pages

Using and Porting GNU CC - 2 GNU CC Command Options
Note, that this **trace** is not the same, as the sequence written to `bbtrace.gz'. It is solely used for counting jump frequencies. -fprofile-arcs: **Instrument** ...
sunsite.ualberta.ca/Documentation/Gnu/gcc-2.8.1/html_chapter/gcc_2.html - 241k - Cached - Similar pages

[GZIP] From ahl at eng.sun.com Sun Apr 1 17:49:21 2007 From: ahl at eng ...
File Format: Gzip Archive - View as HTML
You can use prstat -L -p JVM_pid to select an interesting LWP ID (**thread** ID). When I wrote this, I found that I had a lot of errors like "**scratch space** ...
mail.opensolaris.org/pipermail/dtrace-discuss/2007-April.txt.gz - Similar pages

avr-g++
They all invoke a **trap handler** for one of these instructions, and then the **trap** ..... including the mini- mum amount of **scratch space** recommended by IBM. ...
ccrma.stanford.edu/planetccrma/man/man1/avr-g++.1.html - 356k - Cached - Similar pages

[PDF] F -G D I C O S K A T A dissertation submitted in partial ...
File Format: PDF/Adobe Acrobat
expense of a stack back-**trace** per blocked **thread** on each sample were not ..... trap or illegal instruction, which will transfer control to a **trap handler**. ...
pages.cs.wisc.edu/~tamches/mydissertation.pdf - Similar pages

[PDF] DYNAMIC INSTRUCTION STREAM EDITING Marc Corliss Computer and ...
File Format: PDF/Adobe Acrobat
cluding a hierarchal implementation of a memory region **tracking** table and an adaptation. of **thread**-level speculation for serializing the function call ...
www.elewis.net/papers/MCorlissThesis.pdf - Similar pages

Man page for gcc
They all invoke a **trap handler** for one of these instructions, and then the **trap** ..... including the minimum amount of **scratch space** recommended by IBM . ...
www.doc.ic.ac.uk/lab/labman/lookup-man.cgi?gcc(1) - 396k - Cached - Similar pages

1 2 3 4 5 **Next**
